

ALBERI BINARI DI RICERCA- ESERCIZI

PIETRO DI LENA

DIPARTIMENTO DI INFORMATICA – SCIENZA E INGEGNERIA
UNIVERSITÀ DI BOLOGNA

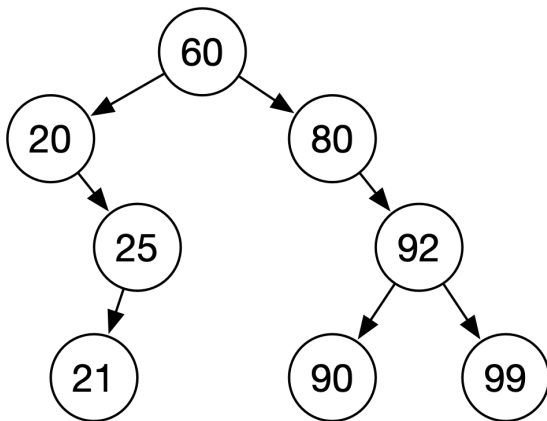
ALGORITMI E STRUTTURE DI DATI
ANNO ACCADEMICO 2021/2022



ESERCIZIO 1

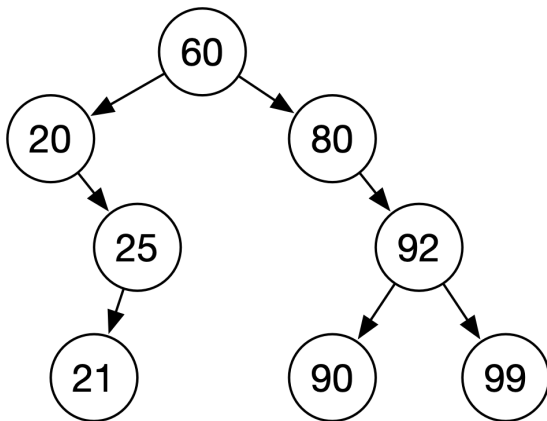
- Dato un Albero Binario di Ricerca con chiavi numeriche intere, inizialmente vuoto, disegnare l'albero ottenuto dopo l'inserimento in ordine dei seguenti valori: 60, 80, 20, 25, 92, 21, 99, 90

ESERCIZIO 1 - SOLUZIONE

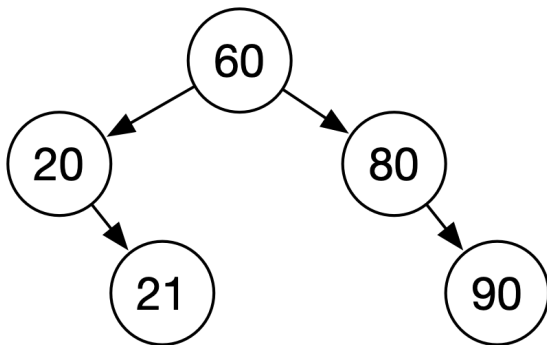


ESERCIZIO 2

- Cancellare dall'albero ottenuto nell'esercizio 1 (mostrato sotto) i seguenti nodi in ordine: 92, 25, 99

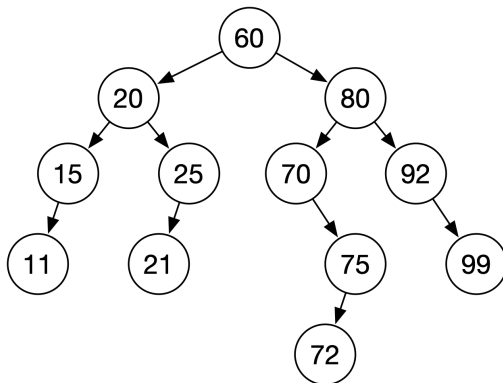


ESERCIZIO 2 - SOLUZIONE

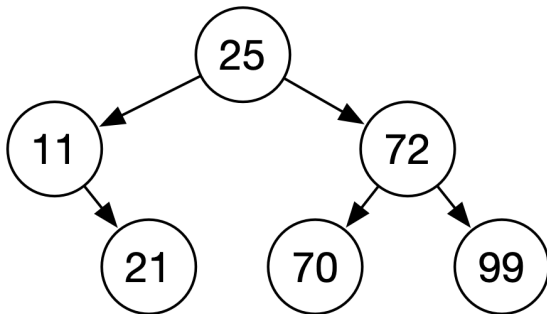


ESERCIZIO 3

- Cancellare dall'albero mostrato sotto i seguenti nodi in ordine:
80, 15, 20, 75, 60, 92



ESERCIZIO 3 - SOLUZIONE



Esercizio 4

- E' vero che se un nodo in un BST ha due figli, allora il suo successore non ha un figlio sinistro e il suo predecessore non ha un figlio destro?
- Giustificare la risposta

Esercizio 4 - Soluzione

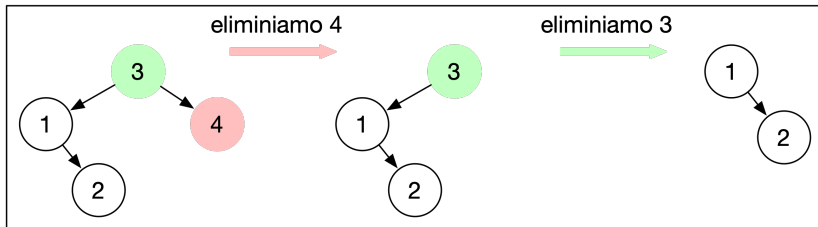
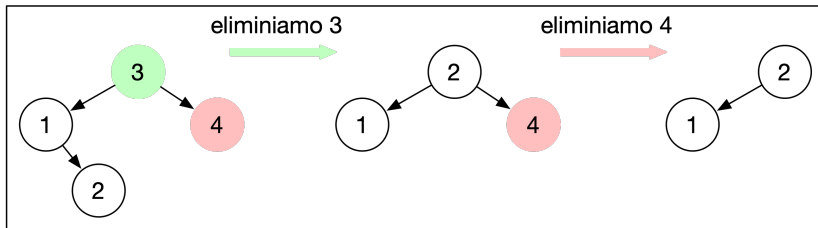
- Assumiamo che u abbia due figli
- Il predecessore di u è il nodo v che viene visitato per ultimo in una visita in-ordine del sottoalbero sinistro di u ($u.left$)
 - Nel caso in cui le chiavi siano tutte distinte v coincide con il nodo con chiave massima nel sottoalbero sinistro di u
- Assumiamo che v sia il *nodo massimo* in $u.left$ e che abbia un figlio destro, allora in una visita in-ordine v non può essere l'ultimo nodo ad essere visitato e questo contraddice l'ipotesi che v sia il nodo massimo. Quindi v non può avere un figlio destro.
- Questa dimostrazione si applica simmetricamente al caso successore

Esercizio 5

- L'operazione DELETE su BST è commutativa?
- Per esempio, dato un qualsiasi BST, eliminare prima un nodo u e poi un nodo v produce sempre lo stesso BST che otterremmo eliminando prima v e poi u ?
- Giustificare la risposta

Esercizio 5 - Soluzione

- Mostriamo con un contro-esempio che l'operazione DELETE su un BST non è commutativa



ESERCIZIO 6

- Dato un albero binario di ricerca, scrivere un algoritmo ricorsivo che stampi i valori delle chiavi in ordine decrescente

ESERCIZIO 6 - SOLUZIONE

```
1: function REV-INORDER(BST  $T$ )  
2:   if  $T \neq \text{NIL}$  then  
3:     REV-INORDER( $T.\text{right}$ )  
4:     PRINT( $T.\text{key}$ )  
5:     REV-INORDER( $T.\text{left}$ )
```

Costo: $\Theta(n)$, n = numero di nodi in T

ESERCIZIO 7

- Scrivere un algoritmo efficiente che dato in input l'albero binario di ricerca T e due valori interi a e b , con $a < b$, ritorni il numero di nodi la cui chiave appartiene all'intervallo $[a, b]$ (estremi inclusi)

ESERCIZIO 7 - SOLUZIONE

```
1: function COUNT(BST  $T$ , INT  $a$ , INT  $b$ )  $\rightarrow$  INT
2:   if  $T == \text{NIL}$  then
3:     return 0
4:   else if  $T.\text{key} < a$  then
5:     return COUNT( $T.\text{right}$ ,  $a$ ,  $b$ )
6:   else if  $T.\text{key} > b$  then
7:     return COUNT( $T.\text{left}$ ,  $a$ ,  $b$ )
8:   else
9:     return 1 + COUNT( $T.\text{left}$ ,  $a$ ,  $T.\text{key}$ ) + COUNT( $T.\text{right}$ ,  $T.\text{key}$ ,  $b$ )
```

- Costo nel caso ottimo (nessuna chiave in $[a, b]$): $O(h)$, h = altezza di T
- Costo nel caso pessimo (tutte le chiavi in $[a, b]$): $\Theta(n)$, n = nodi in T